PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-233326
(43)Date of publication of application : 10.09.1993

(51)Int.Cl. G06F 9/46

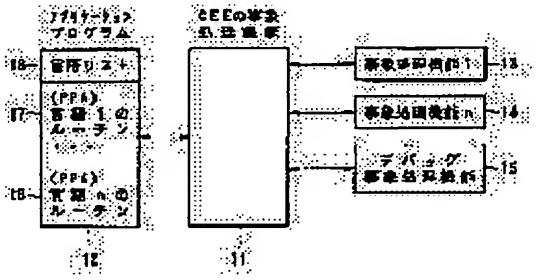
(21)Application number : 04-295194
(22)Date of filing : 04.11.1992

(71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>
(72)Inventor : CARNEY WILLIAM PETER
ENGLAND LAURENCE EDWARD
HOCHMUTH GARY JOHN
OWINGS BRIAN
PORTER ERIC LYNN
SHANNON ALFRED WILLIAM
WILSON ROBERT AARON

(30)Priority
Priority number : 91 810619 Priority date : 19.12.1991 Priority country : US

(54) METHOD AND SYSTEM FOR HANDLING EVENT IN COMPUTER SYSTEM

(57)Abstract:
PURPOSE: To provide a method and system for handling an event with a related parameter at the time of executing a program.
CONSTITUTION: In order to handle the events within a computer system, which is generated in the middle of executing the program 12 including routines in plural programming languages, the number and identification of respectively unique program languages are decided by a language list 16 or another equivalent means. Unique event handling means (event processing functions) 13 to 15 are initialized by each unique programming language and at the time of executing the program, the selected event of the event processing function is detected and its parameter is decided. The detected event is divided into a multi-address communication or desired events, a multi-address communication events (except for a debugging event processing function) are sent to all the event processing functions 13 and 14, and a desired event is event sent to one event processing function. An event code and the related parameter are sent to the event processing functions 13 to 15. These event processing functions 13 to 15 generate a proper returning record supporting the success, failure, not-processing of the event and returns necessary information to each selected event.



LEGAL STATUS

[Date of request for examination]	04.11.1992
[Date of sending the examiner's decision of rejection]	16.09.1997
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平5－233326

(43)公開日　平成５年(1993)９月10日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/46	３４０　Ａ	8120－5B		

審査請求　有　　請求項の数９(全 18 頁)

(21)出願番号　特願平4－295194

(22)出願日　平成４年(1992)11月４日

(31)優先権主張番号　８１０６１９

(32)優先日　1991年12月19日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
INTERNATIONAL BUSINESS MASCHINES CORPORATION
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　ウィリアム、ピーター、カーニー
アメリカ合衆国カリフォルニア州、サン、ノゼ、カレロ、ヒルズ、コート、7153

(74)代理人　弁理士　頓宮　孝一　（外５名）

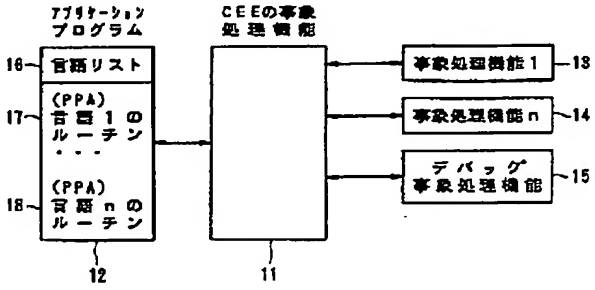
最終頁に続く

(54)【発明の名称】　コンピュータシステムにおいて事象を取り扱う方法及びシステム

(57)【要約】　（修正有）

【目的】　プログラム実行時、関連するパラメータを有する事象を取り扱うための方法と装置の提供。

【構成】　複数のプログラム言語のルーチンを含むプログラムの実行中生じるコンピュータシステム内で事象を取り扱うため、各独自のプログラム言語の数及び識別は言語リストまたは他の等価手段で決定される。独自の事象取り扱い手段（事象処理機能）は各独自のプログラム言語毎に初期化され、プログラム実行時に、イベント処理機能の選択された事象の検出と、そのパラメータが決定される。検出された事象は同報通信または目標に分割され、同時通信事象は（デバッグ事象処理機能を除く）すべての事象処理機能に送られ、目標事象は１つのイベント処理機能に送られる。事象コード及び関連パラメータは、事象処理機能に送られる。この事象処理機能は、事象の成功、失敗または非処理を支持する適当な戻りコードの発生と、選択された事象毎に必要な情報を戻す。



1

【特許請求の範囲】

【請求項１】関連するパラメータを有しかつ複数のコンピュータプログラム言語で準備されたルーチンを含むプログラムの実行中に発生するコンピュータシステム内の事象を取り扱うための方法であって、

プログラムの準備中に使用される独特のコンピュータプログラムを決定するステップと、

独特の各コンピュータプログラミング言語用の独特の事象取り扱い手段であるイベント処理機能を初期化するステップと、

プログラムの実行中選択されたイベントの発生を検出するステップと、

選択されたイベントの関連するパラメータを決定するステップと、

識別事象コード及び関連するパラメータを少なくとも１つの事象処理機能に渡すステップとを有するコンピュータシステムにおいて事象を取り扱う方法。

【請求項２】共通の実行環境を有するコンピュータシステム内で事象処理機能を操作する方法であって、

（ａ）共通の実行環境から事象コード及びパラメータを受け取るステップと、

（ｂ）事象が事象処理機能によって活動を要求するか及び共通の実行環境に対して“アクションが生じない”ことを応答するかどうかを決定するステップと、

（ｃ）事象が事象処理機能から情報を必要とし、必要な情報を見つけるために応答するかどうかを決定するステップと、

（ｄ）ステップ（ｂ）または（ｃ）内で取り扱われない事象を処理するステップと、

（ｅ）ステップ（ｃ）または（ｄ）で実行されたアクションの成功、非処理または障害をステップ（ｃ）で発見された情報とともに共通の実行環境に戻すステップとを有するコンピュータシステムにおいて事象処理機能を操作する方法。

【請求項３】複数のコンピュータプログラム言語で準備されたルーチンを含むプログラムの実行中発生する事象を取り扱うようになっているコンピュータシステムであって、

プログラムの準備に使用される独特のコンピュータプログラム言語を決定するための手段と、

各独特のコンピュータプログラム言語の独特の事象取り扱い手段である事象処理機能を初期化するための手段と、

選択された事象に関連するパラメータを決定するための手段と、

事象コードと関連するパラメータを少なくとも１つの処理機能に通過させるための手段とを有するコンピュータシステム。

【請求項４】選択された事象が同報通信タイプかどうかを決定する手段と、

2

すべての同報通信タイプ事象において、初期化されたすべての事象処理機能に関連するパラメータ及び識別事象コードを送る手段とを有する請求項３に記載のシステム。

【請求項５】プログラムプロローグ領域から取られた高水準言語メンバ識別子を使用することによって事象コード及びパラメータを送る事象処理機能を決定するための手段を有する請求項３に記載のシステム。

【請求項６】初期化手段はあらかじめ割り当てられたメンバ識別子を使用して各事象処理機能用に割り当てられる名称を形成する手段と、

割り当てられる名称を使用して各事象処理機能の動的ロードを要求する手段とを有する請求項１に記載のシステム。

【請求項７】事象の処理の成功、非処理または障害を示す事象処理機能から戻りコードを受け取る手段を有する請求項３に記載のシステム。

【請求項８】ＤＳＡの所有権、エントリポイント及びコンパイルユニット識別、ステートメント識別またはＤＳＡクラシィフィケーションを表す事象処理機能からの情報を受け取る手段を有する請求項３に記載のシステム。

【請求項９】デバッグ事象処理機能を初期化する手段を有する請求項３に記載のシステム。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明はコンピュータシステムのプログラム実行管理の分野の方法及び装置に関する。

【０００２】

【従来の技術】コンピュータプログラミング用の種々の高水準言語（ＨＬＬｓ）（例えば、Ｃ，ＣＯＢＯＬ，ＦＯＲＴＲＡＮ，ＰＬ／Ｌ，ｅｔｃ）は異なる走行時間事象処理の要求がある。１つの言語で形成される事象は他の言語で定義されない。

【０００３】典型的なプログラム言語で書かれたプログラムの実行は、特定の走行時間サポートコードが存在し、各コンパイラによって生じたコードが正しく作用することができるように初期化される。走行時間サポートはすべてのプログラムがそのプログラムで書かれていない場合に数多く要求される種々の機能を含む。入力／出力操作に対するルーチンは１つの例である。

【０００４】いくつかの言語で書かれたルーチンからプログラムをつくることができることが望ましい。従って、走行時間サポートは、言語の任意の要求を異なる要求で同時に満たすことができなくてはならない。多数の実行時間サポートパッケージに現れる共通のルーチンを分離し、それを複数の言語のアプリケーションプログラムをサポートする共通の走行時間支持パッケージに配置し、それによって、言語の特定のサポートコンポーネントのサイズを減少させ及び冗長性を減少させることができる。

3

【０００５】共通の実行環境は、（「メンバ」と呼ばれる）ＨＬＬが種々のリソースを管理することができるバーチャルマシンインタフェースを確立する。このようなバーチャルマシン内において、「事象を無視すること」を含むメンバによって取られるマンデートアクションを取る種々の重要なイベントが生じる。バーチャルマシン内のこのような重要な事象の発生のメンバを通知し、メンバが適当なアクションをとることを可能にする一様な方法が必要になる。さらに、バーチャルマシンは、時間の別々の点でメンバから情報を得て、種々の活動を実行するためにメンバに要求する矛盾のない方法が必要になる。

【０００６】

【発明が解決しようとする課題】従来の技術において、事象の取り扱いのためにベースの走行時間及びＨＬＬの特定のコンポーネントの間の異なる特定のインタフェースの類似した知識が必要になる。従来技術によって提供されない必要なものは、異なる高水準言語ルーチンを１つの走行時間環境で交互に作動させることができる、プログラム実行中の事象の取り扱いを管理するための方法及び手段である。

【０００７】

【課題を解決するための手段及び作用】本発明は複数のコンピュータプログラム言語で準備されたルーチンを含むプログラムの実行中生じるコンピュータシステム内で事象を取り扱うための方法及び手段を開示する。プログラムの準備において使用される各独自のコンピュータプログラム言語の数及び識別は言語リストまたは他の等価手段を使用して決定され、特定の事象取り扱い手段（事象処理機能）が各独自のコンピュータプログラム言語毎に初期化される。プログラムが実行されるときに、イベント処理機能に関する選択された事象が検出される。選択される事象に関するパラメータは決定される。さらに検出された事象は２つのタイプ、すなわち同報通信または目標に分割される。同時通信事象は（デバッグ事象処理機能を含むまたは含まない）すべての事象処理機能に送られ、それ故、目標事象は１つのイベント処理機能に送られ、事象コード及び関係するパラメータは１つのイベント処理機能に送られる。事象コード及び関連パラメータは、各事象処理機能が、アクションがサポートされるプログラミング言語の文脈の事象に適当ならばなんでも実行するように事象処理機能に送られる。この事象処理機能は、事象の成功、失敗または非処理を支持する適当なリターンコードを発生し、及び選択された事象毎に必要な情報を戻す。本発明は分離した特定のデバッグ事象処理機能を提供する。

【０００８】

【実施例】次の用語はこの明細書に使用される。

【０００９】ＣＡＡ

共通のアンカ領域（ＣＡＡ）はアプリケーションの実行

4

スレッドの現在の状態を表す制御ブロックである。

【００１０】ＣＩＢ

処理される最後の状態例えば、エラー、例外等に関する情報を含む制御情報ブロック。

【００１１】コンディション

この用語コンディション及びエクセプションは、時折相互変換的に使用される。これらのコンディション及びエクセプションは、無効なマシンのアドレスまたはゼロによって割算する試みのようなエラーである。またコンディションは実際のエラーではないが、特定のアテンションを必要とする事象を含む。

【００１２】コンディション・マネージャ

はコンディションが生じたときコンピュータシステムの制御を得、種々のシステム及び／または使用者のアプリケーションのルーチンを実行することによってコンディションの取り扱いを管理するプログラムまたは手段である。

【００１３】ＤＳＡ

ダイナミックセイブエリア、例えばスタックフレーム

ＥＤＢ

このＥｎｃｌａｖｅ　Ｄａｔａ　Ｂｌｏｃｋ（ＥＤＢ）はエンクレーブの現在の状態を表し、このエンクレーブによって保持されるリソースを表す制御ブロックである。

【００１４】エンクレーブ

（エンクレーブ）Ｅｎｃｌａｖｅは１グループの手順の実行をサポートする論理的な走行時間構造である。このエンクレーブ内に呼び込む第１の手順は「主な」手順として公知であり、他のものは「サブ」手順として公知である。このエンクレーブは割り当てられた記憶装置及び端子を含む高水準言語の意味の範囲を制限する。

【００１５】フィードバック　コード

フィードバックコードは状態またはエラーの包含（ｅｎｃａｐｓｕｌａｔｉｏｎ）である。

【００１６】ヒープ　ストレージ

プロセス内で走行するプログラムに関するプログラム記憶領域の発生セグメントの順列化されていないグループ。すなわち、特定のタイプのプログラム記憶領域は発生セグメントの獲得及び配置の論理的な順序がない、またｈｅａｐ内のプログラム記憶領域の発生セグメントが特定されないｈｅａｐとして見ることができる。

【００１７】イノベーション

イノベーションは論理的な呼び出し／戻り機構によって制御を有するコード「手順」の実行例である。

【００１８】ＯＣＢ

このオプションコントロールブロック（ＯＣＢ）は走行時間オプションの現在の設定を含む制御ブロックである。

【００１９】ＰＣＢ

このＰｒｏｃｅｓｓ　Ｃｏｎｔｒｏｌ　Ｂｌｏｃｋ（Ｐ

5

ＣＢ）は処理の現在の状態を表し、また処理水準で保持されるリソースを表す制御ブロックである。

【００２０】プラットフォーム

プラットフォームは操作システムとプログラムが実行されるコンピュータのハードウエアとの組み合わせである。

【００２１】プロセデュア

変化する意味がなく別々にコンパイルされることのできる最も小さいソースシーケンスに対応するコンパイラユニット。

【００２２】プロセス

プロセスは特性が実行時間環境によって説明される最も外側の構造である。処理の間のハイクラーキカル（ｈｉｃｒａｒｃｈｉｃａｌ）な関係がない。むしろ、他の顕著な論理的に別れた「アドレススペース」によって各処理はいくつかのシステムのリソースにおいて独立して同期的に競争する。処理は１つまたはそれ以上のエンクレーブを有する。

【００２３】ＰＰＡ

プログラム・プロローグ・エリアは標準のコンパイラによって発生し、コンパイラユニットに関する情報を含む。

【００２４】スタック・フレーム

スタック・フレームはスタック（ｓｔａｃｋ）の１つのエレメントである。スタック・フレームは、手順が呼び出しのために戻るときに手順を呼び出す及び削除される各時間につくられる。手順呼び出し、実行及び戻りに関するリソースを管理するために使用される。

【００２５】スタック・フレーム・ゼロ

スタック・フレーム・ゼロは第１のルーチン用のスタックフレームの直前の概念的なスタックフレームである。第１の手順は、スタック・フレーム・ゼロから呼び出され、終了はスタック・フレーム・ゼロから初期化される。状態の取り扱いの目的でゼロ番目のスタック・フレームは言語用の省略アクションが加えられるフレームである。

【００２６】スタティック・ストレージ

このプログラムはエンクレーブ内でプログラムが走行するとき第１回目の新しい割り当てを得、プログラムが戻るときに自動的に再び割り当てられないプログラムの作動記憶装置。スタティックストレージはエンクレーブ内のプログラムの複数の走行にわたって持続する。

【００２７】スレッド

スレッドは走行時間環境のプログラムモデル内での実行の基本的ユニットである。それが所有するリソースは機械の状態、スタック及び論理的に明瞭な状態の管理者である。各スレッドは実行毎に同期的に競争する。

【００２８】本発明は、種々の高水準言語コンパイラ及び共通の実行環境（ＣＥＥ）の事象取り扱いユニットに関する特定の走行時間事象取り扱いモジュールの間のインタフェース内で実行される。図１は、本発明に基づいたシステムのブロック構成要素を示す。ＣＥＥ１１の事象取り扱いユニットはアプリケーションプログラム１２と事象処理機能１３－１５との間をインタフェースする。このＣＥＥ及び事象処理機能は操作システムから別れたプログラムとして問題の状態で走行するが、必要とされる基本的な変更なしに操作システムの一部として容易に含まれ得る。（単に通常プログラムと称されるアプリケーションプログラムはプログラムの種々の構成要素の準備に使用されるすべてのＨＬＬを識別する言語リスト１６を含む。各ルーチン（コンパイラユニット）１７－１８は書き込まれる言語リストの指示を含む。これはプログラムプロローグエリア（ＰＰＡ）に割り当てられたメンバ数を配置することによって行うことが望ましい。

6

【００２９】図２は、ＥＨＵの主な構成要素を示す。この言語リストプロセッサ２１はアプリケーションプログラムの内部の言語リストの内容を見つけこれを解釈するようになっている。この出力は付加ライブラリから必要な事象処理機能を負荷するために事象処理機能イニシャライザ２２によって使用されるメンバのリストである。事象処理機能イニシャライザは事象ディテクタ２３が初期化が完了した後アクティブになることができるようにする。関連する事象の発生セグメントが事象ディテクタによって検出されたとき、事象処理機能インタフェースユニット２４は必要なパラメータを集め、１つまたはそれ以上の事象処理機能に事象のタイプを識別する事象コードとともにそれらを送る。また、事象処理機能インッターフェイスユニットは事象処理機能によって供給される戻りコード及び戻り情報を受ける。この事象の検出は公知の従来技術の手段によって実行される。典型的には、この事象はプログラムの活動で明示される。例えば、プログラムが終了するとき、適当な事象処理機能を駆動するＣＥＥを通知する。暗黙の事象はプログラムの実行中に起こるエラーまたは他の状態に関連する。

【００３０】定義された事象が起こるときに、正しい事象処理機能にそれを経路指定するために決定の数をつくらなければならない。事象の基本的なタイプは、もし事象処理機能に送られるならば、どのパラメータを送るべきか、及び事象がすべての事象処理機能に関するかどうか、または１つの処理機能のみを呼び出すべきかを決定する。事象が現在実行中のルーチンに対応する事象処理機能にのみ関することが決定されるならば、その事象処理機能の識別はメンバコードを含む現在のルーチンのＰＰＡを検査することによって確かめられる。事象処理機能に順に送られる事象はいわゆる同報通信と呼ばれる。本発明の特定の実施例は、どの事象が同時通信に必要か必要でないかを決定する。各事象処理機能は、特定の事象が呼び込まれるとき活動をしないオプションを有する。ＰＰＡがナンバーコードを含まないならば、目標と

される事象は、事象の所有者を見つけだすまで同時通信である。図３は本発明によるＥＨＵによって実行される主なステップを示す。プログラムに使用される言語の識別はプログラムに含まれる言語リストから決定され（ステップ３１）、事象処理機能は各言語毎に初期化される（ステップ３２）。関連事象が検出されると（ステップ３３）、関連パラメータが決定される（ステップ３４）。事象を同時通信するかどうかに関する決定が行われ（ステップ３５）、もしそうなら、事象コード及びパラメータは事象処理機能に送られる。各事象処理機能は少なくとも戻りコードに戻り適切に処理される（ステップ３６）。非同時通信事象において、正しい事象処理機能の識別は現在実行中のＰＰＡに含まれるメンバ数から決定される（ステップ３７）。事象コード及びパラメータは識別された事象処理機能に送られ（ステップ３８）、戻り情報は情報処理機能が完了した後に処理される（ステップ３９）。

【００３１】本発明は、一般化された環境と個々のＨＬＬの特定の環境とを分離し、事象処理機能とデバッガ事象処理機能のそれぞれを介してデバッグを行う。事象処理機能によって提供されるインタフェースは、特定のＨＬＬまたはデバッガの親密な知識を有するためにＣＥＥを必要とすることなく追加の言語及びデバッガがＣＥＥで走行することができるようにする。

【００３２】本発明は、バーチャルマシン内でメンバの存在を積極的に識別するための機構を与え、バーチャルマシン内でバーチャルマシンからメンバに通信する手段を確立し、この重要な事象のメンバ通知し、この事象の通信とメンバからの応答のインタフェースを定義し、メンバが、発生する重要な事象のために所望の活動を実行するフレームワークを与え、バーチャルマシンにメンバから情報を提供する機構を提供することによって作動する。

【００３３】関係するＨＬＬメンバとしてＣＥＥと適当にインタフェースするためにメンバはロード可能なモジュールの形態で事象処理機能を提供しなければならない。ＣＥＥは標準のオペレーティング・システム機能及び実行可能なコードの開始アドレスを使用してロードされる事象処理機能は、オペレーティング・システムによってＣＥＥに戻される。これは種々の重要な事象の発生セグメントと通信し、ＣＥＥがメンバから情報を受け取る機構を提供する。特定の部材数は、メンバコードとして１バイトのみが使用されるから、好ましい実施例では２５６に任意に制限される。

【００３４】図４はＣＥＥによって呼び出されるときに事象処理機能によって取られる必要があるアクションのフローチャートを示す。イベントコード及びパラメータはインタフェースユニットから受け取られなければならない（ステップ４１）。この最初の決定はこの特定の事象処理機能によってすべてのアクションを必要とするか

に関して行われる。４の戻りコードはアクションが起こらないときに使用される。いくつかの事象がＣＥＥに供給される戻り情報を必要とするから、これらの事象は別に取り扱わなければならない（ステップ４５）。残りの事象は特定の言語に適当な従来の技術によって処理される（ステップ４６）。成功または失敗の戻りコードは、もしあれば、要求された情報とともにＣＥＥの戻される（ステップ４７）。

【００３５】対話式デバッガはメンバが関する事象の組から明瞭である１組の事象に関する。重要な事象のこの通信はデバッガイベント・ハンドラとして知られている別の事象処理機能によって行われる。特に、デバッガイベント・ハンドラは関係するデバッガによって提供されなければならない。多数のデバッガが利用可能であるが、関連するデバッガイベント・ハンドラを有する利用可能な唯１つの利用可能なデバッガが特定のプログラムの実行によって使用される。

【００３６】言語リスト

ＣＥＥはプログラム内に収容された言語リストを検査することによって環境の初期化中どのメンバが存在するかを決定する。これは次のステップによって達成される。

【００３７】１．各メンバは固定されたフォーマット名を有する「記号ＣＳＥＣＴ」を提供しなければならない。

【００３８】２．１組のメンバ記号ＣＳＥＣＴ名用の弱い外側基準のベクトルが確立される。

【００３９】３．メンバがアプリケーション（例えば、ロードモジュール）内にあるならば、標準のリスト編集処理は言語リスト内の割り当てられたスロット内のメンバの記号ＣＳＥＣＴ用のアドレスを配置するが、メンバがプログラム内にないならばメンバの記号ＣＳＥＴはインデックスはゼロの値を含む。メンバの存在が検出されるときメンバの事象処理機能はＣＥＥによってバーチャルメモリにダイナミックにロードされる。弱い外側リストを含む言語リスト用のサンプルＣＳＥＣＴは図５に見られる。

【００４０】事象処理機能

この事象処理機能は重要なイベントが起こったとき、またはメンバによって保持されたいくつかの情報をＣＥＥが必要とするときにプログラムの実行を通じて何回か呼び出されるメンバ供給ルーチンである。この事象処理機能は、メンバ言語の必要な親密な知識を含むが、ＣＥＥは本発明の使用を通じてこのタイプの詳細を知る必要がない。

【００４１】ＣＥＥの初期化中、ＣＥＥはこのアプリケーションに存在するメンバの組を決定する。これらのメンバの存在毎に、ＣＥＥは事象処理機能を負荷する。事象処理機能の名称は固定されたプレフィックス及びメンバ部材を結び付けることによって任意につくられる。つくられた名称は、「ＣＥＥＥＶｘｘｘ」であり、ｘｘｘ

9

はメンバ部材である。この名称は、標準の操作システム呼び出しによって事象処理機能を含むモジュールを負荷するために使用される。事象処理機能のアドレスは、後の検索のために保管される。メンバ数は任意であるが、好ましい実施例において、メンバ用の言語リストのスロットに対応している。

【００４２】ＩＢＭシステム／３７０のリンケージコンベンションを使用して、メンバ事象処理機能へのリンケージは標準のパラメータアドレスリストを含む「ＢＡＬＲ１４．１５」及びＲ１を介して行われる。この第１のパラメータは通常事象処理機能を呼んだ事象のタイプを指示する。追加のパラメータは特定の事象による。

【００４３】種々のダンプサービスを処理する間、事象処理機能はダンプ事象コード７とともに呼び出される。ダンプ事象コードはどのダンプサービスを実行するかを

10

表す関数コードのパラメータを有する。ダンプ事象用の残りのパラメータは特定の関数コードによって変化する。

【００４４】この言語ユーティリティ事象６はどの情報を要求するかを説明する関数コードパラメータを有する。ユーティリティ用の残りのパラメータは特定の関数コードによって変化する。

【００４５】事象のリスト及びそれらの対応するパラメータを表１に示す。ＣＥＥＩＮＴはＣＥＥを初期化するＣＥＥ内の呼び出し可能な入力点用のシンボルラベルである。ＣＥＥＳＴＡＲＴは、負荷モジュールを独特に指定し、言語リストのような情報用の負荷モジュールの経路選択を行うためのシンボルラベルである。

【００４６】

【表１】

11　　12

表　１　　事象コード及びパラメータ

事象	コード	パーム２	パーム３	パーム４	パーム５	パーム６	同報通信
処理	１	ＣＩＢ	結果	新しい状態			NO
イネーブル処理	２	ＣＩＢ	結果	新しい状態			NO
SFOCond処理	３	ＣＩＢ	結果	新しい状態	０		NO
オプションProc	４	オプションブロック	ceestartのアドレス	INPL	512バイトワークエリヤ		NO
Mainopts	５	INPL	R13CEEINTへのインバウンド	R0CEEINTへのインバウンド	R1CEEINTへのインバウンド	Main opts	NO
プロセスInit	１７						YES
言語ユーティリティ	６	関数コード					NO
Dumpサービス	７	ファンクションコード					YES
GOTOターゲットDSA	１０	ターゲットDSA					NO
DSAExitルーチン	１１	開始ブロックのdsa					NO
Enclave Create	１８	プログラムマスク返却	INPL	メンバー特定スレッドトークンまたは０			YES
Enclave終了	１９	INPL					YES
Process終了	２１						YES
デバッグInfo	１６						NO
ATTERM事象	１５						NO
新しい言語モジュール事象	８	負荷モジュールエントリポイント	CEESTARTまたは０				NO

表１の最後のコラムは事象がすべてデバッグ事象機能に対して非同報通信すべきかを指示する。本発明の特定の実施例は、どの事象が同報通信であるかを決定するが全体のエンクレーブ上の効果を有する事象は一般に同報通信である。これは、処理及びエンクレーブの初期化及び終了に関する事象を含む。

【００４７】目標の事象（非同報通信）が処理されるときに、関連する事象処理機能が決定されなければならない。これは事象が発生したときに実行されたルーチンにおいてＰＰＡを決定することによって行われることが好ましい。言語を識別する独特の方法が実行される。また、正しい事象処理機能を発見するまでユーティリティ事象を連続して同報通信することが可能である。

【００４８】ＣＥＥは、プロセス初期化のために及びエンクレーブ初期化のためにメンバ特定初期化ルーチンを呼び出す。このリソース及び能力は２つの事象の間で異なる。次の事象を例として説明する。

【００４９】・処理初期化
・エンクレーブ初期化
・処理終了
・エンクレーブ終了
・走行時間オプション事象
・ａｔｅｒｍ終了事象

ＣＥＥは通常の呼び出しコンベンションに適応するために戻りにおいてその元の値を記憶するようになっている。事象処理機能は次のような値戻りコードの１つに対してＲ１５内に戻りコードを設定しなければならない。

【００５０】

13　　14

戻りコード	意味
4	この事象のためにアクションが起こらない。
１６	事象の処理が成功せず、及び／またはプログラムを直ちに終了しなければならない。

【００５１】ＣＥＥは事象処理機能が１６の値に戻るか、前のリストにない値に戻るならばプログラムを終了する。

【００５２】処理初期化事象：ＳＨＩＲＩ初期化は事象１７である。この事象は処理水準でＨＬＬ部分を持ち出すために使用する。事象処理機能を呼び出す順序は本発明の部分として定義されないが、そうでなければ強制される。

【００５３】メンバに入力するとき、その呼び出し者のレジスタ及びＲ１を記憶することができるＤＳＡを示す事象処理機能Ｒ１３は、事象コード１７の１つのパラメータを有する標準のＯ／Ｓスタイルプリストのアドレスを含む。

【００５４】事象１７及び事象１８の組み合わせは与えられたアプリケーション用の環境のＨＬＬ特定のアスペクトを初期化しなければならない。この事象のためのカウンターパートは事象２１である。

【００５５】処理終了事象：この処理終了事象コードは２１である。この事象は処理水準でＨＬＬ部分を終了させるために使用される。メンバ事象処理機能を呼び込む順序はこの発明の部分を定義しないが、強制される。

【００５６】このプリストは処理の終了用の事象コードの１つのパラメータを含むＯ／Ｓスタイルのプリストである。

【００５７】この事象はＨＬＬが処理の水準で維持されるすべてのリソースを放棄しなければならないことを示す。アプリケーションを終了させるためのすべてのＨＬＬの意味は事象１９エンクレーブ終了事象によってすでに達成されていることに留意すべきである。

【００５８】この事象用のカウンタパートは事象１７である。

【００５９】エンクレーブ初期化事象：このエンクレーブ初期化事象コードは１８である。この事象はエンクレーブ水準でＨＬＬ部分を初期化するために使用される。メンバ事象処理機能を呼び出す順序は本発明の一部を形成しないが、強制される。すべてのＣＥＥサービスは第１の事象の時間に利用可能である。このメンバは以下に説明するようにプログラムマスクの要求を第２のパラメータに配置することによってプログラムマスク設定に影響を与える。エンクレーブ初期化事象用のメンバ事象処理機能に入力するときに、次のものが利用可能である。

【００６０】・Ｒ１４，１５は連係レジスタ
・Ｒ１２はＣＡＡを指定する。
・Ｒ１３はＤＳＡを指定する。
・Ｒ１は次のリストとともに（パラメータのすべては基準によって送られる）標準のＯ／Ｓスタイルプリストのアドレスを含む。

【００６１】１．事象コード１８
２．プログラムマスクを最も右に保持するフルワード分野、入力時にこの領域はゼロになる。
３．ＣＥＥＩＮＴへ送られる初期化プリスト（ＩＮＰＬ）
４．（ＣＩＣＳのもとで実行中に）メンバ特定スレッドトークン、またはゼロ
事象１７及び１８の組み合わせは与えられたアプリケーション用の環境のＨＬＬの特定の観点を初期化しなければならない。

【００６２】エンクレーブ終了事象：このエンクレーブ終了事象コードは１９である。この事象は、エンクレーブ水準でＨＬＬの部分を終了させるために使用される。

【００６３】メンバ事象処理機能に入力中、次のものが利用可能である。
・Ｒ１４，１５は連係レジスタ
・Ｒ１３はＤＳＡを指定する。
・Ｒ１２はＣＡＡを指定する。
・Ｒ１は次のリストとともに（パラメータのすべては基準によって送られる）標準のＯ／Ｓスタイルプリストのアドレスを含む。
－エンクレーブ終了１９を指示する事象コード
－ＣＥＥ初期化中にＣＥＥＩＮＴに送られる初期化パラメータリスト
この呼び出しによって、ＨＨＬが終了エンクレーブの言語意味を補強することによってアプリケーションを意味論的に終了させることができる。エンクレーブ関連リソースは解放されなければならない。この事象は事象１８のカウンタパートである。

【００６４】走行時間オプションイベント：走行時間オプションは数字４である。この事象は能力を制限している。利用可能なスタックがなく、ＣＥＥの呼び出し可能なサービスもない。この目的はコンパチブルな態様で走行時間のオプションをメンバが取り扱うことができるようにすることである。このパラメータリストは次のようである。

【００６５】１．事象コード４
２．ＯＣＢのアドレス
３．ＣＥＥＳＴＡＲＴのアドレス
４．「主な」入力点のアドレス
５．５１２バイトのワークエリアのアドレス
アトターム（Ａｔｔｅｒｍ）事象：このアトターム事象はエンクレーブの終了中に呼び出される。それはすべて

のユーザスタックフレームがスタックから取り除かれた後、エンクレーブ終了事象のメンバを呼び出す前に呼び出される。呼び出しを介してＣＥＥのＣＥＥＡＴＴＲＭサービスに登録されたメンバのみが呼び出される。この事象に送られるパラメータリストは１つのパラメータ、事象コード１５からなる。

【００６６】ダンプ事象処理機能：ダンプ事象処理機能は次のサンプル手順状態に示すパラメータでつくられる。

【００６７】一般的なパラメータフォーマット

手順（ダンプ　事象　コード　関数　コード［追加　パーム］，ｆｃ）

ここでダンプ　事象　コードは７の値を有するフルワード２進の整数である。ファンクション　コードは実行されるダンプ関数を特定するフルワード２進整数である。それは次の値の内１つを含む。

【００６８】１．なぜそのダンプをとるかについて説明する情報メッセージをダンプする。この関数　コードは、いわゆるＣＥＥ３ＤＭＰが第１の場所内に取られるダンプの結果のエラーメッセージをプリントする言語ライブラリの出口を特定する。この情報メッセージはエラー毎にＭＳＧＦＩＬＥに送られるエラーメッセージのコピーである。これらのメッセージはエラーの時にＡＢＥＮＤコード、プログラム状態ワード及びレジスタの内容を含む。ＣＥＥ３ＤＭＰがメンバ言語ライブラリによって呼び出されなければ、メンバ言語ライブラリはこの出口でメッセージをプリントしない。

【００６９】２．ルーチンのアーギュメントをダンプする。メンバ言語がルーチン毎に利用可能なアーギュメントとローカル変数の間で識別できないならば、それが変数をダンプするダンプサービスによって呼ばれると同じ時間にアーギュメントをダンプしなければならない。

【００７０】３．ルーチンの変数のダンプ。これはルーチンによって使用される局所変数と割り当てられた外部変数を含む。メンバ言語ライブラリはそれが決定されるならばこのルーチンによって使用され、または設定された変数のみをダンプしなければならない。

【００７１】４．ルーチンに関連するダンプ制御ブロック。これはメンバ言語によってマップされたＤＳＡ及びデバッグ用に使用可能なルーチンに関する他の制御ブロックを含む。これは、コンパイラシンボル表及びステートメント表を含む。

【００７２】５．ルーチン用の記憶装置のダンプ。これは、自動スタックフレーム記憶装置静的ローカル変数記憶装置を含む。このルーチンと他のルーチンとの間で割り当てられた静的データ記憶装置は、ダンプされなければならない。割り当てられた記憶領域に１つだけのコピーはダンプされなければならない。

【００７３】６．スレッドに関するダンプ制御ブロック。スレッドのためのＣＡＡはＣＥＥによってダンプされる。

【００７４】７．スレッドに関するダンプ記憶装置。ＣＥＥはスレッドに関するすべてのスタック記憶装置をダンプする。メンバ言語はこの出口を使用するスレッドに関連する他のスタック記憶装置をダンプする。スレッドによって使用されるスタック記憶装置はそれが、たとえそれに関連しなくてもダンプされる。データ記憶装置だけをダンプしなければならない。コード含む記憶装置は可能ならばダンプされるべきではない。

【００７５】８．エンクレーブに関するダンプ制御ブロック。エンクレーブ用のＥＤＢはＣＥＥ並びにメンバリストによってダンプされる。メンバの言語はメンバリストから遮断された通信領域をダンプしなければならない。通常アプリケーション負荷モジュールの部分である静的なライブラリ通信領域がある。

【００７６】９．エンクレーブに関するダンプ記憶装置。メンバ言語はこの出口を使用するエンクレーブに関連する他の記憶装置をダンプする。これは通常操作システム記憶装置管理に対する直接的な呼び出しを介して得られる記憶装置を含む。データ記憶装置のみをダンプしなければならない。コードを含む記憶装置は可能ならばダンプするべきではない。

【００７７】１０．ファイルのダンプステータス及び属性．ＣＥＥはメッセージサービスによって使用されるファイルの状態及び属性をダンプする。メンバ言語はそれら自身の状態及び属性をダンプしなければならない。これは、アプリケーションを実行する途中で現在のオープンファイル並びに前のオープンファイルを含む。

【００７８】１１．ファイルに関するダンプ制御ブロック。ファイルステータスを保持する制御ブロック及び他の言語特定制御ブロックがダンプされる。

【００７９】１２．ファイルに関するダンプ記憶装置バッファ。これらのバッファは動作システムによって割り当てられ、典型的にはＣＥＥヒープサービスを使用しない。ＣＥＥヒープサービスによって割り当てられたバッファ記憶装置がダンプされる。

【００８０】１３．この処理に関するダンプ制御ブロック。処理用のＰＣＢがＣＥＥによってダンプされる。

【００８１】１４．処理に関するダンプ記憶装置。データ記憶装置のみがダンプされなければならない。コードを含む記憶装置はダンプされるべきではない。

【００８２】１５．追加のグローバルな情報をダンプする。この情報は、ダンプリポートの最後に表れる。ロードされたライブラリモジュールのリストは、追加のグローバルな情報の一例である。

【００８３】１６．エンクレーブの変数をダンプ。これはこのエンクレーブによって使用されるすべての静的な外部変数を含む。

【００８４】１７．ダンプ呼び出しの最後。これはダンプの瞬間毎に事象処理機能に対する追加の呼び出しがな

17

いことを指示する。

【００８５】追加の　パームはある関数コードに対する特定のパラメータである。次のダイヤグラムは各関数コードを示す。ダンプ　事象　コード７が常に関数コードを越えることに留意すべきである。

【００８６】関数コードによるパラメータフォーマット

手順（７，１，ｆｃ）

手順（７，２ｄｓａｐｔｒ，ｃｉｂｐｔｒ，ｃａａｐｔｒ，ｃｄｂｐｔｒ，ｆｃ）

手順（７，３ｄｓａｐｔｒ，ｃｉｂｐｔｒ，ｃａａｐｔｒ，ｃｄｂｐｔｒ，ｆｃ）

手順（７，４ｄｓａｐｔｒ，ｃｉｂｐｔｒ，ｃａａｐｔｒ，ｃｄｂｐｔｒ，ｆｃ）

手順（７，５ｄｓａｐｔｒ，ｃｉｂｐｔｒ，ｃａａｐｔｒ，ｃｄｂｐｔｒ，ｆｃ）

手順（７，６ｃａａｐｔｒ　ｅｄｂｐｔｒ，ｆｃ）

手順（７，７ｃａａｐｔｒ　ｅｄｂｐｔｒ，ｆｃ）

手順（７，８ｅｄｂｐｔｒ，ｆｃ）

手順（７，９ｅｄｂｐｔｒ，ｆｃ）

手順（７，１０ｅｄｂｐｔｒ，ｆｃ）

手順（７，１１ｅｄｂｐｔｒ，ｆｃ）

手順（７，１２ｅｄｂｐｔｒ，ｆｃ）

手順（７，１３ｐｃｂｐｔｒ，ｆｃ）

手順（７，１４ｐｃｂｐｔｒ，ｆｃ）

手順（７，１５ｅｄｂｐｔｒ，ｆｃ）

手順（７，１６ｅｄｂｐｔｒ，ｆｃ）

手順（７，１７，ｆｃ）

ｄｓａｐｔｒはＤＳＡのアドレスを含むフルワードの２進整数である。

【００８７】このパラメータはルーチンがＣＩＢを有さないならばゼロである。

ｃｉｂｐｔｒはルーチン用のＣＩＢのアドレスを含むフルワードの２進整数である。

ｃａａｐｔｒはＣＡＡのアドレスを含むフルワードの２進整数である。

ｅｄｂｐｔｒはＥＤＢのアドレスを含むフルワードの２進整数である。

ｐｃｂｐｔｒはＰＣＢのアドレスを含むフルワードの２進整数である。

ｆｃ（出力）

基準によって送られる１２バイトのフィードバック。この出口から３つの状態、実行の成功、ダンプファイルにメッセージを書き込む場合のエラー、またはメンバ言語ダンプ出口が不成功であったことが起こる。

【００８８】事象処理機能ユーティリティ事象：例外ハンドリングを含む種々のＣＥＥサービスは、言語特定関数を実行する。これらを実行するために、ＣＥＥはメンバ言語ユーティリティ出口を介して情報を受け取る。このユーティリティ出口はＣＥＥを通過させ必要な処理を実行するために必要な情報である。それは、事象コード

18

６を使用したメンバ事象処理機能の一部である。これらの出口の各々において事象処理機能への説明及び連係を以下に示す。すべての連係は６の事象コードを有し、次に特別の機能コードが続き、ユーティリティに対して特定のパラメータが続く。

【００８９】ＤＳＡ所有権

この出口のために、メンバ言語は、ＤＳＡがそれに書かれ、またはそれによって所有されたルーチンに関連するかどうかを特定する。この出口はＰＰＡスタイルのエントリを有しないコードの所有者を決定するためにＣＥＥによって使用される。第１にＣＥＥはこのコードがＰＡＡスタイルの出口を含むかどうかを見るためにチェックする。呼び出し者のＤＳＡの保管されたレジスタのアイキャッチャはそれがＣＥＥエントリ点を示すかどうかを決定するためにチェックされる。これが真実でないならば、ＣＥＥは言語が所有権を宣言するまでＤＳＡの所有権のためにメンバ言語出口を呼ぶ。

【００９０】パラメータフォーマット

手順（６，１ｄｓａｐｔｒ）

ここで：ｄｓｐｔｒ（入力）は活動的なＤＳＡまたはセーブ領域のためにフルワードのポインタである。所有権（出力）は次のものを含むフルワードの２進の整数である。

０　ＤＳＡに対応するソースコードはメンバ言語の中にはない。

１　ＤＳＡに対応するソースコードはメンバ言語の中にある。

入力点及びコンパイラユニット識別

この出口のために、メンバ言語は、入力点名、入力アドレス、コンパイラユニット名、コンパイラユニットアドレス及びルーチン用現在の指示、ルーチンと関連する与えられたＤＳＡ及びＣＩＢを識別する。この出口はルーチンがＰＰＡスタイルの入力を有しないときのみ呼ばれる。

【００９１】パラメータフォーマット

手順（６，２ｄｓａｐｔｒ，ｅｉｂｐｔｒ，コンパイラ　ユニット　ネーム，コンパイラ　ユニット　アドレス，エントリ　ネーム，エントリ　アドレス，コール　インストラクション　アドレス）

ここで、ｄｓａｐｔｒ（入力）は活動的なＤＳＡまたはセーブエリアに対するフルワードポインタである。

ｃｉｂｐｔｒ（入力）

は１つが存在すれば現在の状態用にＣＩＢに対してフルワードポインタであり、このパラメータはゼロである。

【００９２】コンパイラ　ユニット　ネーム（出力）

はＤＳＡに関連するルーチンを含むコンパイルユニットの名前を含むために任意の長さの固定長文字ストリングである。コンパイルユニットの名前を決定することができないならば、このパラメータはすべてのブランクに設定するべきである。このコンパイルユニットの名前が供

給されたストリング内に嵌合することができなければ、それは切り捨てられるべきである。

【００９３】コンパイル　ユニット　名称　長さ（出力）

は入力時にコンパイラユニット名称の長さ及び出口にストリング内に配置されたコンパイラユニット名称の長さを含むフルワード２進整数である。コンパイラユニット名称を決定することができないならば、このパラメータはゼロに設定される。ストリングが有するこの最大長さは２５６バイトである。

【００９４】コンパイラ　ユニット　アドレス（出力）

はコンパイラユニットの始めのアドレスを含むフルワード２進整数である。このコンパイラユニットのアドレスを決定することができないならば、このパラメータはゼロに設定されなければならない。

【００９５】エントリ　名称（出力）

はＤＳＡに関するルーチンにエントリ点の名称を含むための任意の長さの固定長文字ストリングである。エントリ点名称を決定することができないときは、このパラメータはすべてブランクに設定されなければならない。エントリ点名称が供給されたストリング内に嵌合することができないときは、切り捨てられるべきである。

【００９６】エントリ　名称　長さ（出力）

は入口にエントリ点の長さ及び出口でストリング内に配置されたエントリ点の実際の長さを含むフルワード２進整数である。コンパイラユニット名称を決定することができないならば、このパラメータはゼロに設定される。ストリングが有するこの最大長さは２５６バイトである。

【００９７】エントリ　アドレス（出力）

はエントリ点のアドレスを含むフルワードの２進整数である。エントリアドレスを決定することができないならば、このパラメータはゼロに設定すべきである。

【００９８】呼び出し　命令　アドレス（出力）

はルーチンの外側に制御を転送する指示のアドレスを含むフルワードの２進整数である。これは制御がプログラムの中断によって転送されるならば、ＢＡＬＲまたはＢＡＳＳＭのような指示を呼び出し指示のアドレスか、または中断指示のアドレスである。このアドレスを決定することができないならば、このパラメータはゼロに設定されなければならない。

【００９９】ステートメント識別

この出口のために、メンバ言語はルーチンへのステートメントナンバが与えられた指示アドレスとエントリアドレスを識別する。また、ルーチン用のＤＳＡのアドレス及びルーチン用のＣＩＢのアドレスが送られ、この場合、現在のレジスタの内容がステートメントナンバを決定するために必要とされる。

【０１００】パラメータフォーマット

手順（６，３エントリ　アドレス，呼び出し　指示　アドレス，ｄｓａｐｔｒ，ｃｉｂｐｔｒ，ステートメント　ｉｄ，ステートメント　ｉｄ　長さ）

ここで、

エントリ　アドレス（入力）

はルーチンへの入力点のアドレスを含むフルワードの２進整数である。

呼び出し　指示　アドレス（入力）

は識別されるステートメントの指示のアドレスを含むフルワードの２進整数である。

【０１０１】これは、それ自身ＤＳＡ（例えばフェッチグルーコード）を有しない小さいルーチン内の指示のアドレスであることに留意すべきである。このような場合、小さいルーチンは、ルーチンと呼ばれるステートメント用のコードの延長として考慮される。このような場合、メンバ言語は小さいルーチンの呼び出し者のステートメントの数を送り返さなければならない。

【０１０２】ｄｓａｐｔｒ（入力）

は、ルーチン用のＤＳＡのアドレスを含むフルワードのポインタである。ｃｉｂｐｔｒ（入力）は現在の状態のＣＩＢのアドレスを含むフルワードのポインタである。ＣＩＢがない場合には、このパラメータはゼロである。

【０１０３】ステートメント　ｉｄ（出力）

呼び出し　指示　アドレスによってポイントされた指示の状態識別子を含む任意の長さの固定長文字である。このステートメントを決定することができないならば、このパラメータはすべてブランクに設定すべきである。ステートメントｉｄが供給されたストリングの範囲内に嵌合されなければ、切り捨てられなければならない。

【０１０４】ステートメント　ｉｄ　長さ（出力）

は入力時にエントリ点の長さ及び出口にストリング内に配置されたエントリ点の実際の長さを含むフルワード２進整数である。コンパイラユニット名称を決定することができないならば、このパラメータはゼロに設定される。ストリングが有するこの最大長さは２５６バイトである。

【０１０５】ＤＳＡクラシフィケーション

この出口において、メンバ言語は、この手順に関連するＤＳＡのタイプを識別する。

【０１０６】パラメータフォーマット

手順（６，４ｄｓａｐｔｒ，ｃｌａｓｓ）

ここで、

ｓａｐｔｒ（入力）

はＤＳＡまたはセーブエリアのアドレスを含むフルワードポインタである。ｃｌａｓｓ（出力）は送られたＤＳＡのクラシフィケーションを指示する基準によって送られた固定バイナリ（３１）である。次に示すものは戻ったフルワードのフォーマットである。コンパイルされたコードからライブラリコードを識別するために迅速にチェックすることができ、必要ならば、コンパイル／ライブラリのタイプをメンバが修飾することができる。例え

ば、ＰＬ／Ｉが開始ブロック、Ｏｎユニット、手順を識別することができる。

【０１０７】ｘ′ａｂｃｄｙｙｚｚ′

ここでｚｚはX′ＦＦ′のマックス（ｍａｘ）を有するメンバｉｄである。ｙｙはコンパイルされたコードタイプまたはライブラリコードタイプを修飾するためにメンバによって使用される。

ｄはライブラリコードならば１

コンパイルコードならば２である。

【０１０８】戻りの値はＤＳＡの所有者と独自に関連しなければならない。実際に割り当てられた値は任意であり、本発明に影響することなく交換され得る。例えば、X′００００１０００５はＣＯＢＯＬ、コンパイラライブラリルーチンに振り分けられ、X′００００２０００５′はＣＯＢＯＬ、コンパイルされたコードに使用される。

【０１０９】デバッガ事象処理機能：このデバッグ事象処理機能は名称「ＣＥＥＶＤＢＧ」を有するＣＥＥによって負荷可能でなければならない。使用されるデバッガの使用は、ＬＯＡＤに対してＣＥＥ用の装置にその名称を表すことによって実行時間で行われる。ロードの障害は、このプログラムの実行中にデバッガが利用できないＣＥＥに表示する。名称ＣＥＥＥＶＤＧＢはロードライブラリ内で独特である必要があるだけである。異なるロードライブラリで複数のデバッグ事象処理機能がある。プログラムの特定の実行用に１つだけが使用される。

【０１１０】このデバッガ事象処理機能は次の内１つが発生するときにロードされ初期化される。

【０１１１】初期コマンドストリングまたはＰＲＯＭＰＴが発見され及びＴＥＳＴ走行時間オプションは有効である。

【０１１２】エラー状態は第１の時間用に上昇し、ＴＥＳＴ走行時間オプションは特定されたエラーサブオプションとともに有効である。

【０１１３】状態は第１の時間において上昇し、ＴＥＳＴ走行時間オプションは特定されたＡＬＬサブオプションとともに有効である。

【０１１４】ＣＥＥＴＥＳＴに対する呼び出しはＴＥＳＴ走行時間オプション設定に関係なく行われる。

【０１１５】ＣＥＥはデバッガ事象処理機能を介して事象のデバッガを通告する。このパラメータリストは表２に定義される。

【０１１６】

【表２】

表２　デバッガ／ＣＥＥ事象処理機能インターフェイス

デバッガ事象	デバッガ事象コード	パーム２	パーム３	パーム４
上昇した状態	１０１	ＣＩＢ	結果コード	
ｇｏｔｏ	１１１	ｄｓａ		
エンクレーブinit	１１８	クリエータのｅｄｂ		
エンクレーブターム	１１９			
デバッガターム	１２１			
スレッドｉｎｔ	１２０	クリエータのｔｃｂ		
スレッドターム	１２２			
外部エントリ	１２３	++ｄｓａ	cmdストリング	ＩＮＰＬ
モジュールロード	１２４	ｄｓａ	モジュールデスクリプタ	
モジュールデリート	１２５	ｄｓａ	モジュールネーム	
記憶装置フリー	１２６	記憶装置	記憶装置長さ	
状態プロモート	１２７	ＣＩＢ	結果のコード	
状態ｇｏｔｏ	１２８	ｄｓａ		
アテンション	１２９			
デバッガプログラムチェック	１３０	結果コード		
メッセージリダイレクト	１３１	msg_テキスト	ｄｄｎａｍｅ	
CALL CEETEST	１３２	++ｄｓａ	cmdストリング	

表２に使用された用語毎に、次の定義が加えられる。
モジュール　ネーム
削除されたモジュールネームのハーフワード・プレフィクスト・ストリングモジュール・デスクリプタはロードされたモジュールを説明する構造。その構造は次のようである。
【０１１７】ｄｃｌ　１　モジュール　デスクリプタ
３　ロードポイントポインタ
３　固定されたモジュールサイズ
３　エントリポイントポインタ
３　モジュールネームｃｈａｒ（２５５）
結果コード
生じる状態管理者用の固定された（３１）２進値。
【０１１８】記憶装置長さ
記憶装置のバイト数を含む固定された（３１）２進値。
【０１１９】ｃｍｄストリング
デバッガ命令を含むハーフワードプレフィクスストリング。
ｍｓｇ　ｔｅｘｔ
ＣＥＥメッセージサービスを通して伝達されたテキストのハーフワードプレフィックスドストリング。
ｄｄｎａｍｅ
目標ｄｄｎａｍｅのブランクを有するＣＬ８ストリング、左ジャスティファイ、パッド右を有する。
ＩＮＰＬ
ＣＥＥＩＮＴに送られる初期化パラメータリスト。
ノート
１．すべてのパラメータが基準によって送られる。
２．＋＋ｈｌｌライブラリルーチンｄｓａを意味する要求者のｄｓａはＣＥＥサービスまたは使用者のｄｓａの要求者である。
３．リターンコードはレジスタ１５内に配置される。
・００－成功
・１６－デバッガ内のクリティカルなエラー。再び呼び込まない。
【０１２０】先の明細書を使用することによって、本発

25　　26

明は、標準的なプログラミング及び／またはエンジニアリング技術を使用した本発明が実行される。この結果のプログラムはディスク、ディスケット、メモリカード、ＲＯＭまたは他のメモリ装置に記憶される。実行のために、このプログラムはコンピュータのＲＡＭにコピーされなければならない。コンピュータサイエンスの技術の分野における当業者は、本発明を実行するために適当な一般的な目的または特定の目的のコンピュータハードウエアとソフトウエアとを容易に組み合わせることができる。本発明の好ましい実施例を詳細に説明したが、この実施例に対する変形例及び適用が特許請求の範囲で述べた本発明の観点から離れずに行われることは明らかである。

【０１２１】

【発明の効果】以上説明したように本発明によれば、異なる高水準言語ルーチンを１つの走行時間環境で交互に作動させることができる。

【図面の簡単な説明】

【図１】本発明によるアプリケーションプログラム、共通の実行環境（ＣＥＥ）の事象取り扱いユニット及び事象処理機能の間の関係を示すブロック図。

【図２】本発明によるＣＥＥの事象取り扱いユニットの主な構成要素のブロック図。

【図３】本発明による事象取り扱いユニットによって実行される主なステップによるダイヤグラム。

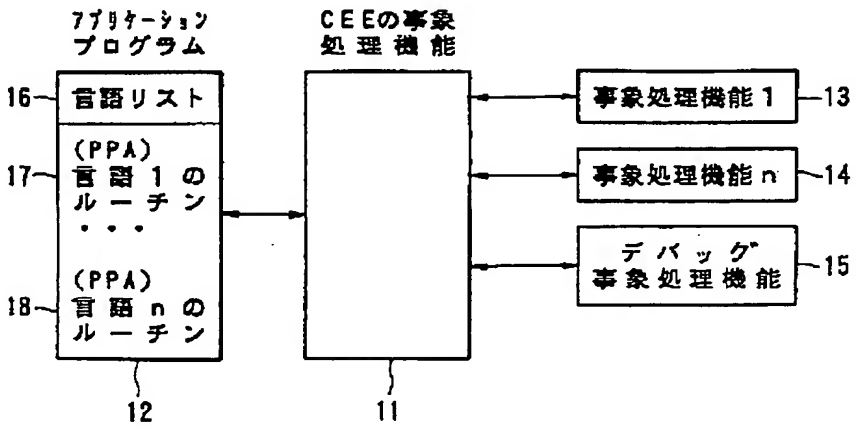
【図４】本発明による事象処理機能のオペレーションにおける主なステップによるダイヤグラム。

【図５】本発明によって使用される言語リストをつくるために使用されるアセンブラ言語ソースコードのサンプルせ説明図。

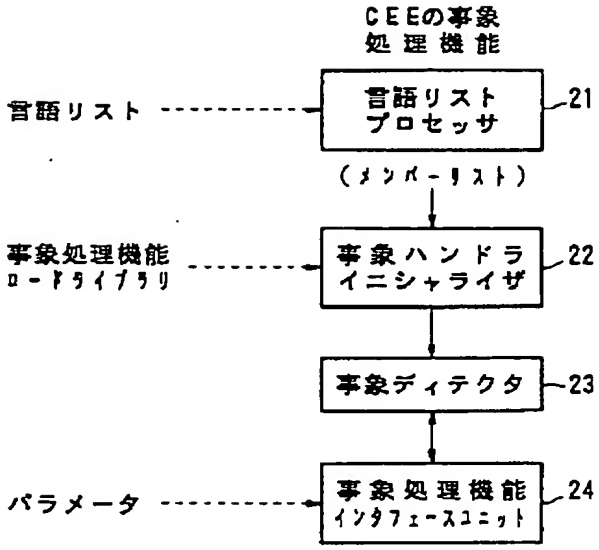
【符号の説明】

１１　ＣＥＥ
１２　アプリケーションプログラム
１３－１５　事象処理機能
１７－１８　ルーチン
２１　プロセッサ
２２　事象処理機能イニシャライザ
２３　事象検出器

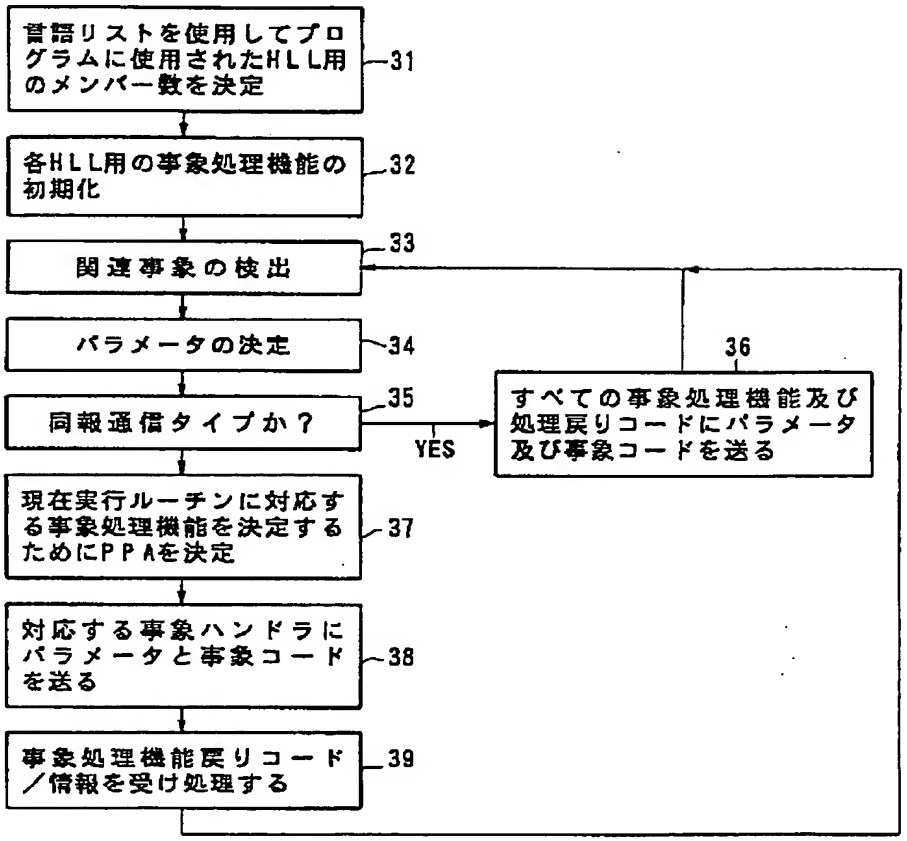
【図１】



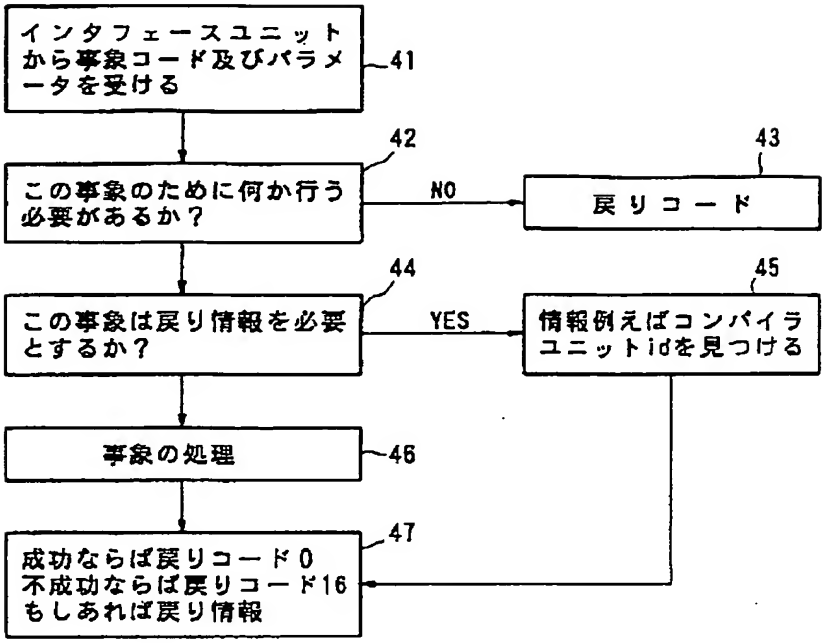
【図２】


CEEの事象
処理機能
言語リスト
言語リスト
プロセッサ
21
（メンバーリスト）
事象処理機能
ロードライブラリ
事象ハンドラ
イニシャライザ
22
事象ディテクタ
23
パラメータ
事象処理機能
インタフェースユニット
24

【図３】


言語リストを使用してプログラムに使用されたHLL用のメンバー数を決定
31
各HLL用の事象処理機能の初期化
32
関連事象の検出
33
パラメータの決定
34
同報通信タイプか？
35
YES
すべての事象処理機能及び処理戻りコードにパラメータ及び事象コードを送る
36
現在実行ルーチンに対応する事象処理機能を決定するためにPPAを決定
37
対応する事象ハンドラにパラメータと事象コードを送る
38
事象処理機能戻りコード／情報を受け処理する
39

【図４】


インタフェースユニットから事象コード及びパラメータを受ける
41
42
この事象のために何か行う必要があるか？
NO
43
戻りコード
44
この事象は戻り情報を必要とするか？
YES
45
情報例えばコンパイラユニットidを見つける
事象の処理
46
47
成功ならば戻りコード0
不成功ならば戻りコード16
もしあれば戻り情報

【図５】

```
CEEBLLST CSECT ,              CEE言語リストヘッダ
CEEBLLST RMODE ANY
CEEBLLST AMODE ANY
         ...
*    実際のリストの開始
         ENTRY CEELLIST
CEELLIST DS    0D             CEE 言語リスト
         WXTRN CEESG000
         DC    A(CEESG000)    00 振り分けられず
         WXTRN CEESG001
         DC    A(CEESG001)    01 コモンサービス
         WXTRN CEESG002
         DC    A(CEESG002)    02 コンパイラ-x
         WXTRN CEESG003
         DC    A(CEESG003)    03 コンパイラ-y
         WXTRN CEESG004
         DC    A(CEESG004)    04 プログラム-x
         WXTRN CEESG005
         DC    A(CEESG005)    05 コンパイラ-z
         WXTRN CEESG006
         DC    A(CEESG006)    06 コンパイラ-j
         WXTRN CEESG007
         DC    A(CEESG007)    07 コンパイラ-k
         WXTRN CEESG008
         DC    A(CEESG008)    08 コンパイラ-l
         WXTRN CEESG009
         DC    A(CEESG009)    09 コンパイラ-m
         WXTRN CEESG010
         DC    A(CEESG010)    10 コンパイラ-n
         WXTRN CEESG011
         DC    A(CEESG011)    11 デバッカ
         WXTRN CEESG012
         DC    A(CEESG012)    12 コンパイラ-o
         WXTRN CEESG013
         DC    A(CEESG013)    13 コンパイラ-p
         WXTRN CEESG014
         DC    A(CEESG014)    14 コンパイラ-q
         WXTRN CEESG015
         DC    A(CEESG015)    15 アセンブラ
         WXTRN CEESG016
         DC    A(CEESG016)    16 コンパイラ-r
         DC    A(0)           ダミーはX'00'を含まなくてはならない
         DS    0D
LLISTEND DC    A(0)           リストの端部のマーク
         END
```

フロントページの続き

(72)発明者　ローレンス、エドワード、イングランド
アメリカ合衆国カリフォルニア州、モーガン、ヒル、ラ、カナダ、コート、520
(72)発明者　ゲーリー、ジョン、ホクマス
アメリカ合衆国カリフォルニア州、サン、ノゼ、モーゼル、ドライブ、6787
(72)発明者　ブライアン、オーウィングス
アメリカ合衆国カリフォルニア州、サン、ノゼ、マカティ、サークル、5703、アパートメント、エイチ

(72)発明者 エリク、リン、ポーター
アメリカ合衆国カリフォルニア州、フリーモント、ノース、モレイ、ストリート、43862

(72)発明者 アルフレツド、ウィリアム、シャノン
アメリカ合衆国カリフォルニア州、モーガン、ヒル、ラ、クロス、ドライブ、830

(72)発明者 ロバート、アーロン、ウィルソン
アメリカ合衆国カリフォルニア州、サン、ノゼ、チェルトナム、ウェイ、51